



به نام خدا



## فاز سوم پروژه کامپایلرها و زبان های برنامه نویسی

پاییز ۹۸

مهلت تحویل: ۲۶ آذر

در این فاز پروژه، قرار است تحلیل معنایی زبان را که در فاز قبلی آغاز کردید به اتمام برسانید. بدین منظور شما قوانین بررسی نوع<sup>۱</sup> را پیاده سازی خواهید کرد. برای تعیین قوانین از اطلاعات بیان شده در ادامه استفاده نمایید.

### گونه:

زبان acton دارای نوع های primitive زیر است:

- int
- string
- int[]
- boolean

علاوه بر این گونه ها، نوع های تعریف شده توسط کاربر نیز هستند که همان اکتور های تعریف شده در برنامه هستند.

همان طور که می دانید هر نوع زیرنوع خودش است (رابطه ی انعکاسی میان نوع ها وجود دارد). برای مثال `int<:int` است. همچنین رابطه ی وراثت یک رابطه ی زیرنوع میان اکتورها ایجاد می کند. برای مثال اگر اکتور B فرزند اکتور A باشد، آنگاه رابطه ی زیر برقرار است:

`B <: A`

<sup>1</sup> Type check

## Overloading

در زبان Acton، تنها عملگرهای `==` و `!=` و تابع `print` به صورت overloaded تعریف شده اند. اگر `e1:T1` و `e2:T2` باشد، آنگاه در عبارت های `e1 == e2` و `e1 != e2`، نوع `e1` و `e2` باید یکسان و از نوع `string`، `boolean`، `int[]` و یا اکتور باشد. در فراخوانی `print(e)`، عبارت `e` می تواند `int`، `string`، `boolean` یا `int[]` باشد. عملگرهای دیگر را طبق جدول زیر بررسی نوع کنید:

عملگرها	گونه ی عملوندها
ضرب (*)، تقسیم (/)، جمع (+)، تفریق (-)، باقی مانده (%)، بزرگتر (>)، کوچکتر (<)، منفی تک عملوندی (-)، تک عملوندی پیشوندی و پسوندی (++ و --)	باید <code>int</code> باشند
عطف منطقی (&&)، فصل منطقی (  )، نقیض منطقی (!)	باید <code>boolean</code> باشند

## فراخوانی msghandler

با توجه به اینکه در این زبان overloading و overriding برای تعریف `msghandler` نداریم، هنگام فراخوانی یک `msghandler` بر روی یک اکتور، آن اکتور باید `msghandler` فراخوانی شده را داشته باشد و تعداد آرگومان های فراخوانی و تعریف یکسان باشد و نوع آن ها زیرنوعی از آرگومان های تعریف شده باشد. (از آنجا که امکان پاس دادن متغیرهای از نوع اکتور نداریم، حالت زیرنوع تنها در حالتی که تایپ `NoType` داریم اتفاق می افتد)

## تخصیص

تخصیص عبارت `e:T` فقط به متغیری از نوع `s` امکان پذیر است که `T<:s` باشد. (در اینجا نیز حالت زیرنوع تنها در دو حالت تخصیص `knownactor` و `NoType` اتفاق می افتد)

## Self

Self همواره نوع اکتوری که در حوزه ی آن فراخوانی شده است را خواهد داشت.

## Statement

Statement ها نیز باید بررسی گونه شوند اما چون مقدار بازگشتی ندارند، گونه در نود آن ها ذخیره نمی شود.

## لیست خطاها

در این فاز لازم است تا در صورتی که در بررسی های فاز دوم خطایی نداشته، موارد زیر را بررسی کنید و در صورت وجود خطا در کد ورودی، پیام مناسب را چاپ کنید. دقت کنید که فاز معنایی شما باید بدون توقف تا انتهای برنامه را بررسی کند و تمامی خطاهای نوع را نشان دهد.

بدین منظور فرض می کنیم که تمامی متغیرهایی که در نوع خود خطا دارند از یک نوع پیش فرض به اسم NoType هستند (NoType زیرنوع تمامی نوع های دیگر است همانند تھی که زیرمجموعه ی تمامی مجموعه های دیگر است).

فرمت کلی هریک از خطاهایی که باید نمایش دهید همانند فاز قبل به صورت زیر است:

Line:<LineNumber>:<ErrorMessage>

### ۱. عدم ارجاع به متغیری که تعریف نشده است

در صورت خطا فرض شود که متغیر وجود دارد و از نوع NoType است.

ErrorMessage: variable <variableName> is not declared

### ۲. بررسی گونه ها هنگام استفاده از عملگرها

در صورت خطا، با استفاده از گونه ی NoType بررسی کد ادامه یابد.

ErrorMessage: unsupported operand type for <operatorName>

### ۳. بررسی گونه ها در ساختارهای تصمیم گیری و تکرار

گونه ی شرط در عبارت if و عبارت وسط for باید از نوع boolean باشد.

ErrorMessage: condition type must be Boolean

### ۴. بررسی وجود تعریف اکتوری که به آن ارجاع داده می شود

این ارجاع ممکن است در هنگام extend کردن باشد، یا هنگام ساخت نمونه از آن اکتور در main برنامه باشد یا در بخش knownactor اکتوری دیگر باشد.

ErrorItemMessage: actor <actorName> is not declared

## ۵. بررسی وجود msghandler در اکتور، در هنگام فراخوانی یک msghandler از یک اکتور

توجه داشته باشید که msghandler ممکن است در خود اکتور موجود نباشد و در اجداد آن اکتور موجود باشد.

ErrorItemMessage: there is no msghandler name <msghandlerName> in actor <actorName>

## ۶. بررسی گونه ورودی تابع print

باید از نوع int، string، boolean یا int[] باشد و نمی‌تواند اکتور باشد.

ErrorItemMessage: unsupported type for print

## ۷. عدم تخصیص به عملوند rvalue

در صورت خطا فرض شود که سمت چپ lvalue است و مشکلی در تخصیص وجود ندارد تا بررسی کد ادامه یابد.

ErrorItemMessage: left side of assignment must be a valid lvalue

## ۸. تطابق تعداد، ترتیب و نوع اکتورهای پاس داده شده به اکتوری دیگر به عنوان knownactor های آن

در اینجا نیز لازم است به زیرنوع بودن knownactor ها دقت شود.

ErrorItemMessage: knownactors does not match with definition

## ۹. اکتور بودن نوع متغیری که msghandler بر روی آن فراخوانی می‌شود

ErrorItemMessage: variable <variableName> is not callable

## ۱۰. استفاده از break و continue داخل حلقه‌ی تکرار

ErrorItemMessage: break/continue statement not within loop

### ۱۱. عدم وجود sender در initial

ErrorItemMessage: no sender in initial msghandler

### ۱۲. lvalue بودن عملوندهای ++ و -- (چه به صورت پیشوندی چه پسوندی)

به طول مثال عبارت ++2 قابل قبول نیست.

ErrorItemMessage: lvalue required as increment/decrement operand

توجه نمایید که لیست بالا برخی از مواردی است که در سند زبان Acton بیان شده و شما باید تمامی موارد داخل سند را پیاده‌سازی کنید.

#### نکات مهم:

- یک فایل با فرمت stdID1\_stdID2.zip (اگر گروهی تک‌نفره هستید stdID.zip) آپلود کنید.
- کد شما باید توسط یک فایل اصلی به نام Acton.java که در کنار کدهای شما وجود دارد، قابل اجرا باشد.